A Trace-Based Semantics for Responsiveness

Walter Vogler*, Christian Stahl† and Richard Müller‡

*Institut für Informatik, Universität Augsburg, Germany; Email: Vogler@informatik.uni-augsburg.de
†Dept. of Mathematics and Computer Science, Technische Universität Eindhoven, The Netherlands; Email: C.Stahl@tue.nl
‡Institut für Informatik, Humboldt-Universität zu Berlin, Germany; Email: Richard.Mueller@informatik.hu-berlin.de

Abstract—In the context of asynchronously communicating services, responsiveness guarantees that a service and its environment have always the possibility to communicate. The responsiveness preorder describes when one service can be replaced by another such that responsiveness is preserved. We study responsiveness for possibly unbounded services with and without final states, and present for both preorder variants a semantical characterization based on traces. Surprisingly, the preorders turn out not to be precongruences, and for both we characterize the coarsest precongruence which is contained in the respective preorder.

Keywords—Petri nets, Asynchronous communication, Precongruence, Should testing

I. INTRODUCTION

Service-oriented computing (SOC) [1] aims at building complex systems by aggregating less complex, independently-developed building blocks called services. A service is an autonomous system that has an interface to interact with other services via asynchronous message passing. Designing a system in such a way allows for rapidly adjusting it to prevalent needs. To this end, services sometimes need to be replaced — for example, when new features have been implemented or bugs have been fixed. This requires a notion of service refinement, which should, according to the idea of SOC, respect compositionality.

Responsiveness means that a service and its environment (called a controller) have always the possibility to communicate. The responsiveness preorder indicates whether a service Spec can be replaced by a service Impl without affecting this property; earlier approaches [2], [3], [4] considered deadlock freedom instead. Several rather ad-hoc variants of responsiveness have been proposed [5], [6], [7], mainly for making the decision of the respective preorders more efficient. Another reason is that controllers can avoid deadlocks by active idling, which is not very reasonable intuitively; so responsiveness was introduced to rule out such controllers, without demanding the rather strict weak termination property (each service has always the possibility to terminate [8]). Usually, controller-based preorders like ours are precongruences and, thus, suitable in the context of SOC; but for responsiveness this was an open question.

In this paper, we investigate this question. Responsiveness has two additional dimensions: the presence of final states and boundedness of service compositions. We present results only for responsiveness for possibly unbounded services with and without final states. For the two preorder variants, we give a semantical characterization based on traces and show that they are no precongruences. Such a characterization helps to understand the respective preorder; it is a basis to determine in concrete cases whether it holds, and (in particular for bounded nets) it can be the foundation for a tool. Then, we characterize the coarsest precongruence which is contained in the respective variant. Here, a characterization is vital, since the definition of a coarsest precongruence is just conceptual and hard to check in concrete cases. The resulting precongruences are close to should testing [9], [10]. We restrict ourselves to unbounded services for two reasons: First, this helps us keeping technicalities simpler. Second, ongoing research already showed that introducing a bound requires only moderate modifications. Section II gives some background; Sect. III introduces responsiveness without final states, characterizes the respective preorder semantically, and presents a characterization of the coarsest precongruence which is contained in this preorder. Section IV lifts these results to the setting with final states. We close with a discussion of related work and a conclusion.

II. PRELIMINARIES

This section provides the basic notions, such as Petri nets, open nets for modeling services, and open net environments for describing the semantics of open nets.

A. Petri nets

As a basic model, we use place/transition Petri nets extended with a set of final markings and transition labels.

For two sets \( A \) and \( B \), let \( A \uplus B \) denote the disjoint union; writing \( A \uplus B \) expresses the implicit assumption that \( A \) and \( B \) are disjoint.

Definition 1 (net). A net \( N = (P,T,F,m_N,\Omega) \) consists of a set \( P \) of places, a set \( T \) of transitions such that \( P \) and \( T \) are disjoint, a flow relation \( F \subseteq (P \times T) \uplus (T \times P) \), an initial marking \( m_N \), where a marking is a mapping \( m : P \to \mathbb{N} \), and a set \( \Omega \) of final markings.

Usually, we are interested in finite nets — that is, nets with finite sets \( P \) and \( T \) — but for some results (e.g., Theorem 20
and 30), we also make use of infinite nets.

Introducing net $N$ implicitly introduces its components $P,T,F,m_0,\Omega$; the same applies to nets $N',N_1$, etc. and their components $P',T',F',m_{N'},\Omega'$, and $P_1,T_1,F_1,m_{N_1},\Omega_1$, respectively — and it also applies to other structures later on.

**Definition 2 (labeled net).** A labeled net $N = (P,T,F,m_0,\Omega,\Sigma_{in},\Sigma_{out},l)$ is a net $(P,T,F,m_{N_1},\Omega)$ together with an alphabet $\Sigma = \Sigma_{in} \cup \Sigma_{out}$ of input actions $\Sigma_{in}$ and output actions $\Sigma_{out}$ and a labeling function $l : T \rightarrow \Sigma \cup \{\tau\}$, where $\tau$ represents an invisible, internal action. Here, we only treat labeled nets where, for every transition $t$, the label $l(t)$ of $t$ is either $\tau$ or $t$ itself.

Graphically, a circle represents a place, a box represents a transition, and the directed arcs between places and transitions represent the flow relation. A marking is a distribution of tokens over the places. Graphically, a black dot represents a token. Transition labels $a \neq \tau$ are written into the respective boxes.

Let $x \in P \cup T$ be a node of a net $N$. As usual, $\cdot x = \{y \mid (y,x) \in F\}$ denotes the preset of $x$ and $x^* = \{y \mid (x,y) \in F\}$ the postset of $x$. We interpret presets and postsets as multisets when used in operations also involving multisets.

A marking is a multiset over the set $P$ of places; for example, $[p_1, 2p_2]$ denotes a marking $m$ with $m(p_1) = 1$, $m(p_2) = 2$, and $m(p) = 0$ for all $p \in P \setminus \{p_1,p_2\}$. We define operators $+, -, =, \leq, \geq$ for multisets in the standard way. We canonically extend the notion of a marking of $N$ to supersets $Q \supseteq P$ of places; that is, for a mapping $m : P \rightarrow N$, we extend $m$ to a marking $m : Q \rightarrow \mathbb{N}$ such that for all $p \in Q \setminus P$, $m(p) = 0$. If $I$ is a set of input places and for all $m \in \Gamma$, $m(p) = 0$, the set $I$ of input places satisfies for all $p \in I$, $m(p) = 0$.

**Definition 4 (open net composition).** An open net $N$ is a tuple $(P,T,F,m_0,\Omega,\Sigma_{in},\Sigma_{out},l)$, such that $I \subseteq \Sigma_{in}$ and $O \subseteq \Sigma_{out}$ if and only if set $I \subseteq \Sigma_{in}$ and $O \subseteq \Sigma_{out}$.

The behavior of a net $N$ relies on the marking of $N$ and changing the marking by the firing of transitions of $N$. A transition $t \in T$ is enabled at a marking $m$, denoted by $m \rightarrow t$, if for all $p \in T^t$, $m(p) > 0$. If $t$ is enabled at $m$, it can fire, thereby changing the marking $m$ to a marking $m' = m - t + t^*$. The firing of $t$ is denoted by $m \xrightarrow{t} m'$; that is, $t$ is enabled at $m$ and firing it results in $m'$.

The behavior of $N$ can be extended to sequences: $m_1 \xrightarrow{t_1} \ldots \xrightarrow{t_{k-1}} m_k$ is a run of $N$ if for all $0 < i < k$, $m_i \xrightarrow{t_i} m_{i+1}$. A marking $m'$ is reachable from a marking $m$ if there exists a (possibly empty) run $m_1 \xrightarrow{t_1} \ldots \xrightarrow{t_{k-1}} m_k$ with $m = m_1$ and $m' = m_k$; for $v = t_1 \ldots t_k$, we also write $m_1 \xrightarrow{v} m_k$. Marking $m'$ is reachable if $m_N = m$. The set $M_N$ represents the set of all reachable markings of $N$.

In the case of labeled nets, we lift runs to traces: If $m_1 \xrightarrow{w} m_k$ and $w$ is obtained from $w$ by replacing each transition by its label and removing all $\tau$ labels, we write $m_1 \xrightarrow{w} m_k$ and refer to $w$ as a trace whenever $m_1 = m_N$. Finally, a net $N$ is bounded if there exists a bound $b \in \mathbb{N}$ such that for every reachable marking $m$ and for all $p \in P$, $m(p) \leq b$.

**B. Open nets**

Like Lohmann et al. [11] and Stahl et al. [2], we model services as open nets [12], [11], thereby restricting ourselves to the communication protocol of a service. In the model, we abstract from data and identify each message by the label of its message channel. An open net extends a net by an interface. An interface consists of two disjoint sets of input and output places corresponding to asynchronous input and output channels. In the initial marking and the final markings, interface places are not marked. An input place has an empty preset, and an output place has an empty postset. We consider only open nets that have either at least one input and one output place or no input and output places; open nets with just input or just output places cannot really take part in a responsive communication.

**Definition 3 (open net).** An open net $N$ is a tuple $(P,T,F,m_0,\Omega,\Sigma_{in},\Sigma_{out},l)$, such that $I \subseteq \Sigma_{in}$ and $O \subseteq \Sigma_{out}$.

The behavior of a net $N$ relies on the marking of $N$ and changing the marking by the firing of transitions of $N$. A transition $t \in T$ is enabled at a marking $m$, denoted by $m \rightarrow t$, if for all $p \in T^t$, $m(p) > 0$. If $t$ is enabled at $m$, it can fire, thereby changing the marking $m$ to a marking $m' = m - t + t^*$. The firing of $t$ is denoted by $m \xrightarrow{t} m'$; that is, $t$ is enabled at $m$ and firing it results in $m'$.

The behavior of $N$ can be extended to sequences: $m_1 \xrightarrow{t_1} \ldots \xrightarrow{t_{k-1}} m_k$ is a run of $N$ if for all $0 < i < k$, $m_i \xrightarrow{t_i} m_{i+1}$. A marking $m'$ is reachable from a marking $m$ if there exists a (possibly empty) run $m_1 \xrightarrow{t_1} \ldots \xrightarrow{t_{k-1}} m_k$ with $m = m_1$ and $m' = m_k$; for $v = t_1 \ldots t_k$, we also write $m_1 \xrightarrow{v} m_k$. Marking $m'$ is reachable if $m_N = m$. The set $M_N$ represents the set of all reachable markings of $N$.

In the case of labeled nets, we lift runs to traces: If $m_1 \xrightarrow{w} m_k$ and $w$ is obtained from $w$ by replacing each transition by its label and removing all $\tau$ labels, we write $m_1 \xrightarrow{w} m_k$ and refer to $w$ as a trace whenever $m_1 = m_N$. Finally, a net $N$ is bounded if there exists a bound $b \in \mathbb{N}$ such that for every reachable marking $m$ and for all $p \in P$, $m(p) \leq b$.

**Definition 4 (open net composition).** Open nets $N_1$ and $N_2$ are composable if $(P_1 \cup T_1 \cup I_1 \cup O_1) \cap (P_2 \cup T_2 \cup I_2 \cup O_2) = (I_1 \cap O_2) \cup (I_2 \cap O_1)$, and $(I_1 \cup I_2) \cap (O_1 \cup O_2)$ and $(O_1 \cup O_2) \setminus (I_1 \cup I_2)$ are both either empty or nonempty.
The composition of two composable open nets $N_1$ and $N_2$ is the open net $N_1 \oplus N_2 = (P, T, F, m_N, I, O, \Omega)$ where

- $P = P_1 \cup P_2 \cup (I_1 \cap O_2) \cup (I_2 \cap O_1)$;
- $T = T_1 \cup T_2$;
- $F = F_1 \cup F_2$;
- $m_N = m_{N_1} + m_{N_2}$;
- $I = (I_1 \setminus I_2) \cup (O_1 \setminus O_2)$;
- $O = (O_1 \setminus I_2) \cup (I_1 \setminus O_2)$;
- $\Omega = \{m_1 + m_2 \mid m_1 \in \Omega_1, m_2 \in \Omega_2\}$.

C. Environments

To give an open net $N$ a trace-based semantics, we consider its environment $\text{env}(N)$ similarly as Vogler [12]. The net $\text{env}(N)$ is a net that can be constructed from $N$ by adding to each interface place $p \in I$ (or $p \in O$) a $p$-labeled transition in $\text{env}(N)$ and renaming place $p$ by $p'$ ($p''$). The net $\text{env}(N)$ is just a tool to define our characterizations and prove our results. But intuitively, one can understand the construction as translating the asynchronous interface of $N$ into a buffered synchronous interface (with unbounded buffers) described by the transition labels of $\text{env}(N)$.

Definition 5 (open net environment). The environment of an open net $N$ is the labeled net $\text{env}(N) = (P \cup P', T \cup I \cup O, F', m_N, \Omega, I, O, l)$ where

- $P' = \{p' \mid p \in I\}$;
- $P'' = \{p'' \mid p \in O\}$;
- $F' = \{(P \times T) \cup (T \cup P)\} \cap F$;
- $F'' = \{(p, t) \mid p \in I, t \in T, (p, t) \in F\}$;
- $F' = \{(t, p') \mid p \in O, t \in T, (t, p) \in F\}$;
- $F'' = \{(p', p) \mid p \in O\} \cup \{(p, p') \mid p \in I\}$;
- $l'(t) = \begin{cases} \tau, & t \in T \\ \in I \cup O. \end{cases}$

The language of $N$ is defined by $L(N) = \{w \in (I \cup O)^* \mid m_{\text{env}(N)} \vdash w\}$.

To compose environments of composable open nets in particular and labeled nets in general, we define a parallel composition $\parallel$ where, for each action $a$ that the components have in common, the $a$-labeled transition of one component is synchronized with the $a$-labeled transition of the other. In addition, we define a second parallel composition operator $\parallel'$. This operator works as operator $\parallel$ and, in addition, hides all common actions — that is, changing the respective labels to $\tau$. Figures 1(d) to 1(f) show the environments of the open nets $L$ and $R$ and their parallel composition. A transition label is depicted inside a transition with bold font to distinguish it from the transition’s identity. The parallel composition with hiding $\text{env}(L) \parallel^h \text{env}(R)$ is as Fig. 1(f), but with transitions $x$ and $y$ labeled $\tau$.

Definition 6 (parallel composition). Labeled nets $N_1$ and $N_2$ are composable if $(P_1 \cup T_1) \cap (P_2 \cup T_2) = (\Sigma_{in_1} \cap \Sigma_{out_1}) \cup (\Sigma_{in_2} \cap \Sigma_{out_2})$. The parallel composition of two composable labeled nets is the labeled net $N_1 \parallel N_2 = (P, T, F, m_N, I, O, l_1)$ and the parallel composition with hiding is the labeled net $N_1 \parallel^h N_2 = (P, T, F, m_N, I, O, l_1)$, where

- $P = P_1 \cup P_2$;
- $T = T_1 \cup T_2$;
- $F = F_1 \cup F_2$;
- $m_N = m_{N_1} + m_{N_2}$;
- $I = \{m_1 + m_2 \mid m_1 \in \Omega_1, m_2 \in \Omega_2\}$;
- $O = \{(m_1 + m_2) \mid m_1 \in \Omega_1, m_2 \in \Omega_2\}$.

The definition of composing two open net environments is a simpler special case compared to, for example, Vogler’s definition in [12], because each label occurs at most once in each component.

To describe the behavior of compositions, we define parallel compositions of words and languages; operator $\parallel$ synchronizes common actions, operator $\parallel'$ also hides them. Observe that in $\text{env}(N_1) \parallel \text{env}(N_2)$ just common transitions are merged; operator $\parallel$ is needed to relate the respective transition sequences.

Definition 7. Given alphabets $\Sigma_1, \Sigma_2$, $\Sigma = (\Sigma_1 \cup \Sigma_2) \setminus (\Sigma_1 \cap \Sigma_2)$, words $w_1 \in \Sigma_1$ and $w_2 \in \Sigma_2$, and languages $L_1 \subseteq \Sigma_1$ and $L_2 \subseteq \Sigma_2$, we define

- $w_1 || w_2 = \{w \in \Sigma_1 \cup \Sigma_2 \mid w|_{\Sigma_1} = w_1, w|_{\Sigma_2} = w_2\}$;
- $w_1 \parallel^h w_2 = \{w \in \Sigma \mid w|_{\Sigma} \subseteq w_1||w_2\}$;
- $L_1||L_2 = \bigcup\{w_1||w_2 \mid w_1 \in L_1, w_2 \in L_2\}$;
- $L_1 \parallel^h L_2 = \bigcup\{w_1 \parallel^h w_2 \mid w_1 \in L_1, w_2 \in L_2\}$.

For composable open nets $N_1$ and $N_2$, let $C = \text{env}(N_1 \oplus N_2)$ and $\bar{C} = \text{env}(N_1) \parallel \text{env}(N_2)$. $C$ and $\bar{C}$ have the same places except for places $p \in (I_1 \cap O_2) \cup (I_2 \cap O_1)$ in $C$ and
the corresponding places $p^i$, $p^o$ in $\Sigma$, yielding a relationship between their reachable markings and their behavior. We use this relation in the proofs by translating operator $\oplus$ to $\uparrow$ and the latter into $\parallel$ followed by hiding of common actions.

Given a set of traces, we define the prefix closure, suffix closure, and the remainder of this set.

**Definition 8 (closures, remainder).** Let $U \in \mathcal{P}(\Sigma^*)$. Then, $\downarrow U = \{ u \in \Sigma^+ \mid \exists v \in U : u \subseteq v \}$ is the prefix closure of $U$, $\uparrow U = \{ u \in \Sigma^+ \mid \exists v \in U : v \subseteq u \}$ is the suffix closure of $U$, and $v^{-1}U = \{ u \in \Sigma^+ \mid vu \in U \}$ is the remainder of $v$ in $U$.

**III. UNBOUNDED NETS AND NO FINAL MARKINGS**

In this section, we consider possibly unbounded open nets and ignore final markings. The resulting notions of responsiveness and $r$-accordance yield an equivalence, which is similar to $P$-deadlock equivalence in [12].

**Definition 9 (responsiveness).** Let $N_1$ and $N_2$ be composable open nets. A marking $m$ of $N_1 \oplus N_2$ is responsive if we can reach from $m$ a marking that enables a transition $t$ with $t^* \cap (O_1 \oplus O_2) \neq \emptyset$. Open nets $N_1$ and $N_2$ are responsive if their composition $N_1 \oplus N_2$ is a closed net and every reachable marking in $N_1 \oplus N_2$ is responsive.

Responsiveness ensures that at least one net can talk to the other repeatedly. This property depends on $N_1$ and $N_2$ in combination: In the composition, $N_1$ ($N_2$) will usually not reach all markings it could reach in other contexts; also, it suffices that just one component can enable an output. In a setting with bounded open nets, this will imply (ongoing work) mutual communication. Based on the correctness criterion responsiveness, we define an $r$-controller of an open net $N$ as an open net $C$ such that $N$ and $C$ are responsive.

**Definition 10 (r-controller).** An open net $C$ is an $r$-controller of an open net $N$ if $N$ and $C$ are responsive.

If the $r$-controllers of an open net are a superset of the $r$-controllers of another open net, then the first open net is a refinement of the second; intuitively, it makes more users happy than the latter. We refer to the resulting refinement relation as $r$-accordance, which gives a necessary requirement for a refinement. For modular reasoning, a refinement relation should be a precongruence for composition. Because $r$-accordance shall turn out not to be one, we will make it stricter (smaller) as far as needed to obtain such a precongruence, and we already introduce a notation for this coarsest precongruence.

**Definition 11 (r-accordance).** For interface equivalent open nets $\text{Spec}$ and $\text{Impl}$, $\text{Impl}$ $r$-accords with $\text{Spec}$, denoted by $\text{Impl} \sqsubseteq_{r,acc} \text{Spec}$, if for all open nets $C$ we have: If $C$ is an $r$-controller of $\text{Spec}$, then $C$ is one of $\text{Impl}$ as well. We denote the coarsest precongruence w.r.t. $\oplus$ contained in $\llbracket_{r,acc}$ by $\llbracket_{r,acc}$.

In the following, we first give a trace-based characterization for $r$-accordance and show that the latter is not a precongruence. Afterward, we characterize the coarsest precongruence contained in the $r$-accordance relation.

A. A trace-based semantics for responsiveness

Our trace-based semantics of an open net $N$ considers the set of stop-traces of its environment $\text{env}(N)$. A stop-trace records a run of $\text{env}(N)$ that ends in a marking weakly enabling actions of $I$ only, such that $N$ stops unless some input is provided. This trace is a weak version of a notion with the same name in [3], [4], where only transitions of $I$ and no $\tau$-transitions are allowed to be enabled.

**Definition 12 (stop-semantics).** Let $N$ be an open net. A marking $m$ of $\text{env}(N)$ is a stop except for inputs if there is no $o \in \Sigma_{out}$ with $m \Rightarrow o$. The stop-semantics of $N$ is defined by the set of traces $\text{stop}(N) = \{ w \mid m_{\text{env}(N)} \Rightarrow w, m \text{ and } m \text{ is a stop except for inputs} \}$.

The presence of stop-traces in open nets $N_1$ and $N_2$ is closely related to the question whether $N_1$ and $N_2$ are responsive.

**Lemma 13 (responsiveness vs. stop-semantics).** Let $N_1$ and $N_2$ be composable open nets s.t. $N_1 \oplus N_2$ is closed. Then $N_1$ and $N_2$ are responsive iff $\text{stop}(N_1) \cap \text{stop}(N_2) = \emptyset$.

Inclusion of the stop-traces of open nets defines a refinement relation which coincides with $r$-accordance. Thus, we obtain a trace-based characterization of $r$-accordance.

**Theorem 14 ($r$-accordance and stop inclusion coincide).** For any interface equivalent open nets $\text{Spec}$ and $\text{Impl}$, we have $\text{Impl} \sqsubseteq_{r,acc} \text{Spec}$ iff $\text{stop}(\text{Impl}) \subseteq \text{stop}(\text{Spec})$.

Proof: $\Leftarrow$: Proof by contraposition. Consider an open net $C$ such that $\text{Impl} \sqsubseteq C$ and, equivalently by interface equivalence, $\text{Spec} \sqsubseteq C$ are closed. Otherwise, $C$ is neither an $r$-controller of $\text{Impl}$ nor of $\text{Spec}$. Assume that $C$ is not an $r$-controller of $\text{Impl}$. Then, $\text{Impl}$ and $\text{C}$ are not responsive by Definition 10, and we find a trace $w \in \text{stop}(\text{Impl}) \cap \text{stop}(C)$ by Lemma 13. Because of stop-inclusion, we have $w \in \text{stop}(\text{Spec}) \cap \text{stop}(C)$. Again with Lemma 13, we see that $\text{Spec}$ and $\text{C}$ are not responsive; that is, $C$ is not an $r$-controller of $\text{Spec}$.

$\Rightarrow$: Let $I$ (O) be the input (output) places of $\text{Impl}$ and, by interface equivalence, of $\text{Spec}$. Let $w \in \text{stop}(\text{Impl})$ and $w = w_1 \ldots w_n$ with $w_j \in I \cup O$, for $j = 1, \ldots, n$. Define open net $N_w = (P,T,F,m_0,O,I,\emptyset)$ with

- $P = \{ p_0, \ldots, p_n \}$, $T = \{ t_1, \ldots, t_n \}$,
- $F = \{(p_i,t_{i+1}) \mid 0 \leq i \leq n - 1 \}$ \cup $\{(t_i,p_i) \mid 1 \leq i \leq n \}$ \cup $\{(w_i,t_i) \mid 1 \leq i \leq n, w_i \in O \}$ \cup $\{(t_i,w_i) \mid 1 \leq i \leq n, w_i \in I \}$, and
\[\begin{align*}
\text{Lemma 13 and Definition 12, there exists } v \in (I \cup O)^* \\
\text{with } m_{\text{env}}(\text{Spec}) \rightarrow m_1 \\
\text{and } m_{\text{env}}(N_{w,o}) \rightarrow m_2 \\
\text{such that both } m_1 \text{ and } m_2 \text{ are stops except for inputs.}
\end{align*}\]

From observation (2), transitions \( t_1, \ldots, t_n \) of \( N_{w,o} \) occur in this order in a run of \( \text{env}(N_{w,o}) \) underlying \( v \) and, thus, there is no occurrence of a transition \( t'_j \) in \( v \) by construction. Furthermore, no transition \( t_w \) has fired and removed the token from \( p \). These facts imply that the Parikh vectors of \( w \) and \( v \) agree: Each \( t_j \) takes a token from or puts a token onto \( w_i \), but all interface places are empty at the end. Each occurrence of \( t_j \) with \( w_j \in t'_j \) (as output place of \( N_{w,o} \); that is, \( w_j \in I \)) is paired with a succeeding occurrence of \( w_j \) (as transition of \( \text{env}(N_{w,o}) \)); otherwise, transition \( w_j \) would be enabled at \( m_2 \) in \( \text{env}(N_{w,o}) \) and \( m_2 \) would not be a stop except for inputs. As transition \( w_j \) is not in conflict with any other transition of \( \text{env}(N_{w,o}) \), we assume that \( w_j \) fires immediately after \( t_j \). In the corresponding rearranged trace \( v' \) of \( v \), all \( w_j \in I \) occur in the same order as in \( w \), and \( v' \) still leads to \( m_2 \).

Similarly, each occurrence of \( t_j \) with \( w_j \in t'_j \) (as input place of \( N_{w,o} \); that is, \( w_j \in O \)) is paired with a preceding occurrence of \( w_j \) (as transition of \( \text{env}(N_{w,o}) \)), which can be delayed such that it occurs immediately before \( t_j \). In the corresponding rearranged trace \( v'' \) of \( v' \), all \( w_j \in O \) occur in the same order as in \( w \), because \( v' \) and \( v'' \) have the same Parikh vector as \( w \); thus, \( v'' \) is \( w \) and \( v'' \) still leads to \( m_2 \).

We have transformed \( v \) to \( w \) by moving \( w_j \in I \) backwards and \( w_j \in O \) forwards. This can also be done in the run underlying \( v \) in \( \text{env}(\text{Spec}) \), because the respective transitions have an empty pre- or postset. Thus, \( m_{\text{env}}(\text{Spec}) \rightarrow m_1 \) and \( m_{\text{env}}(N_{w,o}) \rightarrow m_2 \) and therefore \( w \in \text{stop}(\text{Spec}) \). 

According, as defined in Definition 11, does not guarantee compositionality; that is, it is not a precongruence with respect to open net composition. For a counterexample, see Fig. 3. Open net \( A' \) r-accords with open net \( A \) by Theorem 14, because their stop-traces coincide: Firing of transitions \( t_5 \) and \( t_6 \) after transition \( t_2 \) in \( A' \) can be simulated by transitions \( t_5 \) and \( t_6 \) in \( A \); if a stop except for inputs is reached this way in \( A' \), place \( p_5 \) contains a token; the corresponding marking in \( A \) contains a token in \( p_2 \) and is a stop except for inputs as well. However, \( A' + B \) does not r-accord with \( A + B \): Open net \( C \) is an r-controller for \( A + B \) but not for \( A' + B \), because firing transitions \( t_2 \) in \( A' \) and \( t_2 \) in \( C \) leads to a nonresponsive marking of \( (A' + B) \oplus C \).

The difference of \( A \) and \( A' \) becomes visible if we consider their refusal traces: In \( A \), it is always possible to mark place \( c \) at least once, whereas in \( A' \) an occurrence of transition \( t_2 \) inhibits marking place \( c \). Therefore, it is not possible to differentiate between \( A \) and \( A' \) with something even weaker than standard failure semantics [13] (like the trace semantics we employed in Definition 12).
Lemma 16. For composable open nets $N_1$ and $N_2$, we have $\mathcal{F}^+(\text{env}(N_1 \parallel N_2)) = \mathcal{F}^+(\text{env}(N_1) \uparrow \text{env}(N_2))$. \hfill $\lozenge$

Essentially, the following characterization of the $\mathcal{F}^+$-semantics for the composition of two open nets has been proved for a more general setting with labeled nets in [12]. Lemma 16 makes it possible to reuse this result of [12] here.

Proposition 17 ($\mathcal{F}^+$-semantics for open net comp.). For composable open nets $N_1$ and $N_2$, $\mathcal{F}^+(N_1 \parallel N_2) = \{(w, X) \mid \exists (w_1, X_1) \in \mathcal{F}^+(N_1), (w_2, X_2) \in \mathcal{F}^+(N_2) : w \in w_1 \uparrow w_2 \land \forall x \in X : x \in x_1 \uparrow x_2 \implies x_1 \in X_1 \lor x_2 \in X_2\}$. \hfill $\lozenge$

For the present setting, the tree failures used in the $\mathcal{F}^+$-semantics give too much information about the moment of choice in an open net. This information can be removed by closing up under an ordering over tree failures. The resulting modification of the $\mathcal{F}^+$-semantics yields the following refinement relation.

Definition 18 ($\mathcal{F}^+$-refinement). For interface equivalent open nets $\text{Spec}$ and $\text{Impl}$, $\text{Impl} \sqsubseteq_{\mathcal{F}^+} \text{Spec}$, denoted by $\text{Impl} \sqsubseteq_{\mathcal{F}^+} \text{Spec}$, if $\forall (w, X) \in \mathcal{F}^+(\text{Impl}) : \exists x \in \{\varepsilon\} \cup \downarrow X : (wx, x^{-1}X) \in \mathcal{F}^+(\text{Spec})$. \hfill $\lozenge$

Using $\mathcal{F}^+$-semantics and -refinement is a technically very beneficial reformulation [10] of $\mathcal{F}^+$-inclusion in [12]. Our definition of $\mathcal{F}^+$-refinement is equivalent to the definition of the refinement relation $\sqsubseteq_{\mathcal{F}^+}$ in [10], which coincides with should (or fair) testing [9], [10] as proved in [10, Thm. 36]. Should testing is a precongruence for composition [10], and, with the help of Lemma 16, we can show that it is also a precongruence for composition operator $\oplus$.

Proposition 19 (precongruence). $\mathcal{F}^+$-refinement is a precongruence for operator $\oplus$. \hfill $\lozenge$

With the next theorem, we prove that $\mathcal{F}^+$-refinement and the coarsest precongruence, which is contained in the $r$-accordance relation, coincide.

Theorem 20 (precongruence, $\mathcal{F}^+$-refinement coincide). For any interface equivalent open nets $\text{Spec}$ and $\text{Impl}$, we have $\text{Impl} \sqsubseteq_{\text{r,acc}} \text{Spec}$ iff $\text{Impl} \sqsubseteq_{\mathcal{F}^+} \text{Spec}$. \hfill $\lozenge$

Proof: $\Leftarrow$: In the following, we assume a trace $w \in \text{stop}(\text{Impl})$ and prove $w \in \text{stop}(\text{Spec})$. Then, applying Theorem 14, we get $\text{Impl} \sqsubseteq_{\text{r,acc}} \text{Spec}$, and this in turn also shows the second implication with Proposition 19 and the definition of $\sqsubseteq_{\text{r,acc}}$.

So let $O$ be the set of output places of $\text{Impl}$ and, equivalently by interface equivalence, of $\text{Spec}$. We have $w \in \text{stop}(\text{Impl})$ iff $(w, O) \in \mathcal{F}^+(\text{Impl})$ by Definition 12 and 15. Then, by $\text{Impl} \sqsubseteq_{\mathcal{F}^+} \text{Spec}$, there must be a suitable $x \in \{\varepsilon\} \cup O = \{\varepsilon\} \cup O$ that makes the defining condition of Definition 18 true. We cannot have $x \in O$ because $(wx, (\varepsilon)) \notin \mathcal{F}^+(\text{Spec})$ by Definition 15. Thus, $x = \varepsilon$ and
(w, O) ∈ F+(Spec), implying w ∈ stop(Spec).

⇒: Suppose Impl ⊑r,acc Spec, and let (w, X) ∈ F+(Impl). In addition, consider an open net C with the new output x and the new input y. Open net C has the empty initial marking, no final marking, and contains only a single transition that can indefinitely repeat to produce a token in x while consuming a token from place y. The idea is to construct an open net N from (w, X) such that C is not an r-controller of Impl ⊓ N because of (w, X). By Impl ⊑r,acc Spec and because C ⊑r,acc is a precongruence, we have Impl ⊓ N ⊑r,acc Spec ⊓ N and thus Impl ⊓ N ⊑r,acc Spec ⊓ N by Definition 11. Thus, C is also not an r-controller of Spec ⊓ N, and from this we shall conclude that (w, X) is covered by F+(Spec) according to Definition 18. Then we will have proved Impl ⊑F+ Spec.

The open net N has inputs I = Impl ∪ {x} and outputs O = Impl ∪ {y} and enables a transition sequence v = t1...tk. Each transition in v is connected to an interface place of N such that the corresponding trace of interface actions is w; that is, N contains net Nw as in Fig. 2. Thus, we can essentially fire the trace w of env(N) in Impl ⊓ N and, therefore, in Impl ⊓ N ⊓ C by firing v instead of the labeled transitions. This way, we reach in Impl the marking m that refuses X in env(Impl); in N, there is only one token in a place pe and the token in a place p has been consumed. This token is necessary to enable the transition t' that is essential for responsiveness; that is, transitions t and t' repeatedly communicate with C. The place p can only be marked again by firing some transition t'z with z ∈ X, and this in turn requires the firing of a transition sequence that — similarly to v — looks to Impl like trace z. But this trace cannot be fired at m; hence, C is not an r-controller of Impl ⊓ N.

To achieve the effect just described, the second part of open net N encodes the tree part X of tree failure (w, X). Common prefixes thereby correspond to the same path in the X-part of N. When reaching an element of X, a token can be produced in place p. Figure 4 illustrates this construction; it is a small adaptation of a construction that is used by Vogler in [12, Fig. 3.19].

Let w = w1...wk such that for j = 1,...,k, wj ∈ Impl ∪ OImpl. Define open net N = (P, T, F, mN, O, I, ∅) with

- P = \{p\} ∪ \{pi | 0 ≤ i ≤ k - 1\}
- \{p_i | u ∈ (\downarrow X ∪ \{\varepsilon\})\}
- T = \{t, t'\} ∪ \{ti | 1 ≤ i ≤ k\}
- \{t_u | u ∈ \downarrow X\} ∪ \{t'_z | z ∈ X\}
- F = \{(pi, ti+1) | 0 ≤ i ≤ k - 1\}
- \{(ti, ti) | 1 ≤ i ≤ k - 1\}
- \{(w, ti) | 1 ≤ i ≤ k ∧ w_i ∈ OImpl\}
- \{(t_i, w_i) | 1 ≤ i ≤ k ∧ w_i ∈ Impl\}
- \{(a, t_u) | a ∈ Impl ∧ u ∈ \downarrow X\}
- \{(t_u, u) | a ∈ Impl ∧ u ∈ \downarrow X\}, and
- m_N = (p_0, p).

As argued previously, we now have that C is not an r-controller of Spec ⊓ N; that is, some marking m_1 can be reached in Spec ⊓ N ⊓ C where responsiveness is violated. Clearly, p must be empty in m_1; thus, v has been fired in N plus possibly some transitions in its X-part. There is just one token in the places of inner(N), and it is in some p_a with uu' ∈ X. Let m_2 be the projection of m_1 to the places of Spec. From the point of view of Spec, we have fired a trace uu of env(Spec) reaching m_2. Because in Spec ⊓ N ⊓ C no t'z can become enabled (otherwise, C would be an r-controller), u' cannot be fired in env(Spec) at m_2. Thus, we conclude that Spec, u' ∈ X ∈ F+(Spec) and, therefore, Impl ⊑F+ Spec.

IV. UNBOUNDED NETS AND FINAL MARKINGS

In this section, we consider possibly unbounded open nets with final markings. We refer to the resulting variant of responsiveness as final-responsiveness or f-responsiveness for short.

Definition 21 (f-responsiveness). Let N_1 and N_2 be composable open nets. A marking m of N_1 ⊕ N_2 is f-responsive if either m is responsive or we can reach a final marking of N_1 ⊕ N_2 from m. Open nets N_1 and N_2 are f-responsive if their composition N_1 ⊕ N_2 is a closed net and every reachable marking in N_1 ⊕ N_2 is f-responsive.

The notion of f-responsiveness generalizes responsiveness, as defined in Definition 9. While responsiveness requires at least one net of the composition to repeatedly talk to the other net, f-responsiveness, in addition, also allows...
the composition to terminate — that is, to reach a common final marking.

Next, we redefine the notion of a controller and accordance for this variant of responsiveness.

**Definition 22 (fr-controller, fr-accordance).** An open net \( C \) is an \( \text{fr-controller} \) of an open net \( N \) if \( N \) and \( C \) are \( \text{fr-responsive} \).

For interface equivalent open nets \( \text{Spec} \) and \( \text{Impl} \), \( \text{fr-accords with Spec} \), denoted by \( \text{Impl} \subseteq_{\text{fr,acc}} \text{Spec} \), if for all open nets \( C \) holds: \( C \) is an \( \text{fr-controller} \) of \( \text{Spec} \) implies \( C \) is an \( \text{fr-controller} \) of \( \text{Impl} \). We denote the coarsest precongruence w.r.t. \( \oplus \) contained in \( \subseteq_{\text{fr,acc}} \) by \( \subseteq_{\text{fr,acc}} \).

We continue by first giving a trace-based characterization for \( \text{fr-accordance} \) and showing that the latter is not a precongruence. Thus, we obtain a trace-based characterization for all \( \text{fr-responsive} \) open nets.

### A. A trace-based semantics for final-responsiveness

We extend Definition 12 by a set of dead-traces, a weak version of a notion with the same name in [3], [4]. A dead-trace is a stop-trace leading to a nonfinal stop except for inputs — that is, a marking dead except for inputs.

**Definition 23 (stop-dead semantics).** Let \( N \) be an open net. A marking \( m \) of \( \text{env}(N) \) is dead except for inputs if \( m \) is a stop except for inputs and there exists no final marking \( m' \) of \( \text{env}(N) \) with \( m \Rightarrow m' \). The stop-dead semantics of \( N \) is defined by the sets of traces (1) \( \text{stop}(N) \) and (2) \( \text{dead}(N) = \{ w \mid m_{\text{env}(N)} \Rightarrow w m \text{ and } m \text{ is dead except for inputs} \} \).

The stop-dead semantics can be used to characterize \( \text{fr- responsiveness} \).

**Lemma 24 (f-responsiveness vs. stop-dead semantics).** Let \( N_1 \) and \( N_2 \) be composable open nets such that \( N_1 \oplus N_2 \) is closed. Then \( N_1 \) and \( N_2 \) are \( \text{fr-responsive} \) iff \( \text{stop}(N_1) \cap \text{dead}(N_2) = \emptyset \) and \( \text{dead}(N_1) \cap \text{stop}(N_2) = \emptyset \).

Inclusion of the stop- and dead-traces of open nets defines a refinement relation which coincides with \( \text{fr-accordance} \). Thus, we obtain a trace-based characterization of \( \text{fr-accordance} \). We leave out the proof as it is similar to the proof of Theorem 14.

**Theorem 25 (trace-characterization of fr-accordance).** For any interface equivalent open nets \( \text{Spec} \) and \( \text{Impl} \), we have \( \text{Impl} \subseteq_{\text{fr,acc}} \text{Spec} \) iff \( \text{stop}(\text{Impl}) \subseteq \text{stop}(\text{Spec}) \) and \( \text{dead}(\text{Impl}) \subseteq \text{dead}(\text{Spec}) \).

Like \( r \)-accordance, \( \text{fr-accordance} \) does not guarantee compositionality; that is, it is not a precongruence with respect to open net composition. Consider again Fig. 3 for a counterexample. Open nets \( A, A', B, \) and \( C \) all have no final marking, implying that the stop- and dead-traces coincide for each of these nets. Open net \( A' \) \( \text{fr-accords with open net } A \) by Theorem 25, because their stop- and dead-traces coincide (see explanation after Theorem 14). However, \( A' \oplus B \) does not \( \text{fr-accord} \) with \( A \oplus B \), because open net \( C \) is a \( \text{fr-controller} \) for \( A \oplus B \) but not for \( A' \oplus B \).

### B. Deriving a precongruence for final-responsiveness

The notion of \( \text{fr-responsiveness} \) distinguishes between final and nonfinal markings. This information is needed to determine whether a marking is dead except for inputs. As we cannot derive this information from the \( \mathcal{F}^+ \)-semantics, we must enhance it. The idea is basically to add an additional ingredient to a tree failure \((w, X)\) yielding a triple \((w, X, Y)\). This ingredient is a set \( Y \), collecting traces that cannot lead the net to a final marking — including traces that cannot be performed at all. We bind the traces in \( Y \) to a certain marking \( m \) that is reached by executing \( w \). Different markings \( m \) can be reached by \( w \) because of nondeterminism, so different sets \( Y \) may be assigned to them. This construction ensures that we can identify traces in \( \text{dead}(N) \).

**Definition 26 (\( \mathcal{F}^+_{\text{fin}} \)-semantics).** Let \( N \) be a labeled net with alphabet \( \Sigma = \Sigma_{\text{in}} \uplus \Sigma_{\text{out}} \). The \( \mathcal{F}^+_{\text{fin}} \)-semantics of \( N \) is a set of fintree failures and defined as \( \mathcal{F}^+_{\text{fin}} \)(\( N \)) = \{ (w, X, Y) \in \Sigma^{+} \times P(\Sigma^{+}) \times P(\Sigma^{+}) \mid \exists m \in M_{N} : m_{N} \Longrightarrow w m \land \forall x \in X : m \not\Longrightarrow w x \land \forall y \in Y : \forall m' : m \Longrightarrow m' \implies m' \notin \Omega_{N} \} \).

For an open net \( N \), we define \( \mathcal{F}^+_{\text{fin}}(\text{env}(N)) \). The \( \mathcal{F}^+_{\text{fin}} \)-refinement relation is similarly defined as the \( \mathcal{F}^+ \)-refinement relation in Definition 18 by closing up under an ordering over the fintree failures in \( \mathcal{F}^+_{\text{fin}} \), thereby removing the too detailed information about the moment of choice in an open net.

**Definition 27 (\( \mathcal{F}^+_{\text{fin}} \)-refinement).** For interface equivalent open nets \( \text{Spec} \) and \( \text{Impl} \), \( \text{impl} \mathcal{F}^+_{\text{fin}} \)-refines \( \text{Spec} \), denoted by \( \text{Impl} \subseteq_{\mathcal{F}^+_{\text{fin}}} \text{Spec} \), if \( \forall (w, X, Y) \in \mathcal{F}^+_{\text{fin}}(\text{Impl}) : \exists x \in \{ \varepsilon \} \cup X \cup Y : (wx, x^{-1}X, x^{-1}Y) \in \mathcal{F}^+_{\text{fin}}(\text{Spec}) \).

The next lemma gives the \( \mathcal{F}^+_{\text{fin}} \)-semantics for open net composition. Its proof (left out) uses that Lemma 16 also holds for \( \mathcal{F}^+_{\text{fin}} \) and that operator \( \uparrow \) can be translated to operator \( \parallel \) followed by hiding of common actions.

**Lemma 28 (\( \mathcal{F}^+_{\text{fin}} \)-semantics for open net composition).** For composable open nets \( N_1 \) and \( N_2 \), we have \( \mathcal{F}^+_{\text{fin}}(N_1 \oplus N_2) = \{ (w, X, Y) \mid \exists (w_1, X_1, Y_1) \in \mathcal{F}^+_{\text{fin}}(N_1), (w_2, X_2, Y_2) \in \mathcal{F}^+_{\text{fin}}(N_2) : w \in w_1 \uparrow w_2 \land \forall x \in X, y \in Y : \{ x \in x_1 \uparrow x_2 \implies x_1 \in X_1 \lor x_2 \in X_2 \land (y \in y_1 \uparrow y_2 \implies y_1 \in Y_1 \lor y_2 \in Y_2) \} \).
as follows. First, we show the precongruence result for labeled nets and operator $\parallel$. Then, we show that this result is also preserved under hiding of common actions. Finally, we combine these results to show the precongruence for open nets and operator $\oplus$.

For the first and second step, we can actually reuse the proofs introduced for should testing, in particular in [10, Lem. 46]. This proof is based on saturation conditions like SAT1-3 below. The key idea in [10] is to shift traces from the refusal set of Impl. We apply the same proof strategy for the $X$-part of the fintree failures, which is closed under suffix (SAT3). Because this does not hold for the $Y$-part, we cannot directly apply this idea here. We overcome this problem by adding the set $X$ to the set $Y$, thereby using condition SAT4 on fintree failures.

SAT1 $\langle w, x, y, x', y' \rangle \in F^+_\text{fin}(N), x' \subseteq X, y' \subseteq Y$ implies $\langle w, x', y' \rangle \in F^+_\text{fin}(N)$

SAT2 $\langle w, x, y \rangle \in F^+_\text{fin}(N) \land \forall z \in Z : (wz, z^{-1}X, z^{-1}Y) \notin F^+_\text{fin}(N)$ implies $\langle w, x \cup z, y \cup z \rangle \in F^+_\text{fin}(N)$

SAT3 $\langle w, x, y \rangle \in F^+_\text{fin}(N)$ implies $\langle w, x \cup y \rangle \in F^+_\text{fin}(N)$

SAT4 $\langle w, x, y \rangle \in F^+_\text{fin}(N)$ implies $\langle w, x \cup y \rangle \in F^+_\text{fin}(N)$

SAT1 states that, given a fintree failure $(w, x, y, x', y')$, sets $X$ and $Y$ can be arbitrarily decreased and the resulting triple is again a fintree failure. Furthermore, the refusal part of $F^+_\text{fin}$ is saturated in the sense that sets $X$ and $Y$ can be extended by any set of traces $z$ such that $(wz, z^{-1}X, z^{-1}Y) \notin F^+_\text{fin}(N)$ (SAT2). SAT3 states that the $X$-part is closed under suffix, and SAT4 shows that the refusal part of $F^+_\text{fin}$ is saturated in the sense that set $X$ can be added to $Y$.

Theorem 29 (precongruence). $F^+_\text{fin}$-refinement is a precongruence for operator $\oplus$.

We show that the coarsest precongruence, which is contained in the $fr$-accordance relation, and $F^+_\text{fin}$-refinement coincide.

Theorem 30 (precongruence, $F^+_\text{fin}$-refinement coincide). For any interface equivalent open nets $Spec$ and Impl, we have $\langle Impl \sqsubseteq_{fr, acc} Spec \rangle$ iff $\langle Impl \sqsubseteq F^+_\text{fin} Spec \rangle$.

Proof: $\Leftarrow$: Consider a trace $w \in stop(Impl)$ ($w \in dead(Impl)$); we prove $w \in stop(Spec)$ ($w \in dead(Spec)$).

Then, applying Theorem 25, we get $\langle Impl \sqsubseteq_{fr, acc} Spec \rangle$, and this in turn also shows the second implication with Theorem 29 and the definition of $\sqsubseteq_{fr, acc}$. So let $O$ be the set of output places of $Impl$ and of $Spec$.

We have $w \in stop(Impl)$ iff $\langle w, O, \emptyset \rangle \in F^+_\text{fin}(Impl)$ by Definition 12 and 15. Then, by $\langle Impl \sqsubseteq F^+_\text{fin} Spec \rangle$, there must be a suitable $x \in \{e\} \cup O = \{e\} \downarrow O$ that makes the defining condition of Definition 26 true. We cannot have $x \in O$ because $\langle w, \{e\}, \emptyset \rangle \notin F^+_\text{fin}(Spec)$ by Definition 26.

Thus, $x = e$ and $\langle w, O, \emptyset \rangle \in F^+_\text{fin}(Spec)$, implying $w \in stop(Spec)$.

We have $w \in dead(Impl)$ iff $\langle w, O, \{e\} \rangle \in F^+_\text{fin}(Impl)$ by Definition 23. Again, we find that $x = e$ and thus $\langle w, O, \{e\} \rangle \in F^+_\text{fin}(Spec)$, implying $w \in dead(Spec)$.

$\Rightarrow$: Suppose $\langle Impl \sqsubseteq_{fr, acc} Spec \rangle$, and let $\langle w, x, y \rangle \in F^+_\text{fin}(Impl)$. In addition, consider an open net $C$ with the new output $x$ and the new input $y$. Open net $C$ has the empty initial marking and contains only a single transition that can indefinitely repeat to produce a token in $x$ while consuming a token from place $y$. In addition, its final marking is the empty marking. The idea is to construct an open net $N$ from $(w, x, y)$ such that $C$ is not an $fr$-controller of $Impl \oplus N$ because of $(w, x, y)$. By $\langle Impl \sqsubseteq_{fr, acc} Spec \rangle$ and because $\sqsubseteq_{fr, acc}$ is a precongruence, we have $\langle Impl \oplus N \sqsubseteq_{fr, acc} Spec \oplus N \rangle$ and thus $\langle Impl \oplus N \sqsubseteq_{fr, acc} Spec \oplus N \rangle$ by definition of $fr$-accordance. Thus, $C$ is also not an $fr$-controller of $Spec \oplus N$, and from this we shall conclude that $(w, x, y)$ is covered by $F^+_\text{fin}(Spec)$ according to Definition 27. Then we will have proved $\langle Impl \sqsubseteq_{fr, acc} Spec \rangle$.

The construction of open net $N$ is similar as in the proof of Theorem 20. Open net $N$ has input places $I = O_{Impl} \cup \{x\}$, has output places $O = I_{Impl} \cup \{y\}$, and enables a transition sequence $v = t_1 \ldots t_k$. Each transition in $v$ is connected to an interface place of $N$ such that the corresponding trace of interface actions is $w$; that is, $N$ contains net $N_w$ as in Fig. 2. Thus, we can essentially fire the trace $w$ of $env(N)$ in $Impl \oplus N$ and, therefore, in $Impl \oplus N \oplus C$ by firing $v$ instead of the labeled transitions.

This way, we reach in $Impl$ the marking $m$ that refuses $x$ in $env(Impl)$; in $N$, there is only one token in a place $p_x$ and the token in a place $p$ has been consumed. This token is necessary to enable transition $t'$ that is — together with transition $t$ — essential for $fr$-responsiveness, because they can repeatedly communicate with $C$. Place $p$ can only be marked again by firing some transition $t'_x$ with $x \in X$, and in this turn requires the firing of a transition sequence that — similarly to $v$ — looks to $Impl$ like trace $x$. But this trace cannot be fired at $m$. In addition, every trace $y \in Y$, which cannot lead to a final marking in $Impl$, leads to a final marking of $N$. This construction guarantees that there is a marking reachable in the composition $Impl \oplus N \oplus C$ which is neither responsive (because place $p$ is not marked and hence there is no communication between $C$ and $N$) nor reaches a final marking (because if $N \oplus C$ is in a final marking, then $Impl$ is not). As a consequence, $Impl \oplus N \oplus C$ is not $fr$-responsive and, thus, $C$ is not an $fr$-controller of $Impl \oplus N$.

To achieve the effect just described, the second part of open net $N$ encodes the tree part for $X$ and $Y$ of fintree failure $(w, x, y)$. Common prefixes thereby correspond to the same path in this part. If a path corresponds to some $y \in Y$, a token on the place at the end of this path is a final
marking of $N$. If the path corresponds to some $x \in X$, a token in the respective place allows to mark $p$ again. Again, Fig. 4 illustrates this construction.

Let $w = w_1 \ldots w_k$ such that for $j = 1, \ldots, k$, $w_j \in \text{Imp} \cup O_{\text{Imp}}$. Define open net $N = (P, T, F, m_N, O, I, \Omega)$ as in the proof of Theorem 20, but replace every occurrence of $\downarrow X$ with $(\downarrow X \cup \downarrow Y)$.

As argued previously, we now have that $C$ is not an $fr$-controller of $\text{Spec} \oplus N$; that is, some marking $m_1$ can be reached in $\text{Spec} \oplus N \oplus C$ where $f$-responsiveness is violated. Clearly, places $p$, $x$, and $y$ must be empty in $m_1$; thus, $v$ has been fired in $N$ plus possibly some transitions in the fintree part of the net. There is just one token in the places of $\text{inner}(N)$, and it is in some $p_u$ with $uu' \in X$ (resp. $uu' \in Y$). Let $m_2$ be the projection of $m_1$ to the places of $\text{Spec}$. From the point of view of $\text{Spec}$, we have fired a trace $wu$ of $env(\text{Spec})$ reaching $m_2$. Because in $\text{Spec} \oplus N \oplus C$ no $t_uw$ can become enabled and the composition cannot reach a final marking — otherwise, $C$ would be an $fr$-controller —, no $u'$ can be fired in $env(\text{Spec})$ at $m_2$ and a final marking is not reachable. Thus, we conclude $(wu, \{u' \mid uu' \in X\}, \{u' \mid uu' \in Y\}) \in F_{\text{fin}}^+(\text{Spec})$, and thus $\text{Imp} \subseteq F_{\text{fin}}^+ \text{Spec}$. ■

V. CONCLUSION AND RELATED WORK

We presented two novel semantics for open nets, taking the permanent possibility to communicate (called responsiveness) as a minimal correctness criterion. In the absence of final states, the semantics consists of a set collecting completed traces; in the presence of final states, an additional set is needed to distinguish successfully and unsuccessfully completed traces. We showed that trace inclusion is not a precongruence in either case, and characterized the coarsest precongruence contained in the respective preorder. For the first semantics, this precongruence turns out to be should testing [9], [10]; for the second semantics, it is should testing extended with traces that do not lead to a final marking.

The idea of responsiveness for finite state services with final states has been coined by Wolf [5]. [5] and the less restrictive variant in [7] define responsiveness for single open nets, and excluding all the other nets guarantees something like our responsiveness. In contrast, we are more optimistic and define responsiveness for open net compositions, which allows to deal with more nets. Müller [6] presents an asymmetrical definition from the point of view of one individual service in a composition. Our notion of responsiveness yields an equivalence similar to P-deadlock equivalence in [12].

Other work dealing with the term responsiveness refers to different properties: Reed et al. [14] aim at excluding certain deadlocks, whereas responsiveness in our setting refers to the ability to communicate. Acciai and Boreale [15] want to guarantee communication over specific channels and consider responsiveness in combination with properties such as deadlock freedom and termination.

We are currently working on a setting with bounded open nets, where our notion of responsiveness will imply mutual communication; this might match better with intuition. It is future work to study the relation of our semantics and the compact representation of all controllers in [7].

ACKNOWLEDGMENT

The research of Christian Stahl is supported by the NWO project “Behavior-Oriented Service Substitution.”

REFERENCES